DOCUMENT RESUME

ED 308 071 SE 050 633

AUTHOR McCoy, Leah P.
TITLE Use of Variables: Algebra-Computer-English Translation.
PUB DATE 89
NOTE 16p.; Paper presented at the International Association for Computing in Education Conference (San Francisco, CA, March, 1989).
PUB TYPE Reports - Research/Technical (143) -- Tests/Evaluation Instruments (160)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Algebra; Computer Literacy; *Computer Uses in Education; *Concept Formation; High Schools; Problem Solving; *Secondary School Mathematics; *Symbols (Mathematics); *Test Items

ABSTRACT

Results of empirical research relating computer programming instruction and understanding of the concept of mathematical variables is unclear. While some studies have found a positive relationship, others have reported nonsignificant results. The purpose of this study was to investigate high school students' (n=36) understanding of the concept of variable in computer programming and in algebraic context. High school juniors at a large private high school who had completed Algebra I and a computer literacy course were given the Algebra-Computer-English (ACE) Translation Test. The results of this study indicate that high school students with experience in both algebra and computer programming could work with variables in a computer context better than in an algebraic context. When given parallel items in both a computer program and algebraic context, their correct responses were not uniformly distributed. They were better able to translate to English the variables in the computer program than in the algebraic equation. A list of 30 references is included. An appendix includes the ACE Translation Test. (DC)

Use of Variables: Algebra-Computer-English Translation

Leah P. McCoy

Indiana University at South Bend

Paper prepared for presentation at the

International Association for Computing in Education Conference

San Francisco, CA, March, 1989.

Use of Variables: Algebra-Computer-English Translation

As we progress further into the "computer age", we are faced with increasingly difficult decisions about how we will utilize computers in education. Several years ago, Taylor (1980) classified computer use in schools into three modes: tutor, tool, and tutee. The tutor mode referred to use of computers to teach the child; the tool mode referred to production or management uses; and the tutee mode referred to the computer being taught by the child by programming. Despite the many changes in computer hardware and software, this system of organization is still appropriate.

Much of the early use of computers in schools was the tutee mode, i.e. teaching children to program the computer. More recently, schools have increased computer use in the tutor mode (computer-assisted-instruction), and in the tool mode (applications programs such as word processing). The argument for reducing the amount of computer programming is that very few students will become professional programmers; therefore, there is no necessity of their learning the intricacies of programming. Computer literacy is currently more commonly defined as being able to USE the computer in word processing or as a means to learn or practice other subjects.

The question of whether to teach computer programming to precollege students has been widely debated. One view is that

computer programming experience increases problem solving skills in students (e.g. Mayer, 1975; Nickerson, 1982; Papert, 1980; Shneiderman, 1985). Because computer programming and problem solving are similar tasks and involve similar processes, it seems that there may be transfer of the skills involved. Since programming is itself a problem solving exercise, it seems possible that learning to program a computer may be related to development of general problem solving skills.

Research into the effect of computer programming on general problem solving skill has produced inconclusive results. Some studies have found positive effects of programming experience on general problem solving defined as reasoning skill (Clement & Gullo, 1984; Many, Lockard, Abrams & Friker, 1988; Nowaczyk, 1984; Oprea, 1985; Reed & Palumbo, 1988; Reed, Palumbo & Stolar, 1988), or mathematical ability (Johnson & Harding, 1979; Mayer, Dyck & Vilberg, 1986; McCoy & Orey, 1988). Other studies have found mixed results when examining evidence of a general effect (Linn, 1985; Pea & Kurland, 1984; Turner & Land, 1988).

A major problem with problem solving research is the difficulty in defining and measuring general problem solving skill. Problem solving is a complex process involving a number of distinct and interrelated skills. It seems that we should define and measure these specific components of problem solving instead of "guessing" about effects on the "global" problem solving abilities of students. This would allow us to compare specific skills involved in both computer programming and in

problem solving, and to examine any evidence of transfer.

One specific skill involved in both computer programming and mathematical problem solving is the use of algebraic variables. Understanding of the concept of variable has been identified as an important aspect of both mathematical problem solving (Kantowski, 1982; Polya, 1957) and computer programming (Papert, 1980; Mayer, 1975). Many students, including college students majoring in engineering, have been shown to lack understanding of the concept of variable (Clement, 1982; Clement, Lochhead & Monk, 1981; Fisher, 1988). These studies involved representation of situations as algebraic equalities, the classic "students and professors" problem:

> Write an equation using the variables S and P to represent the following statement: "There are six times as many students as professors at this university." Use S for the number of students and P for the number of professors.

Most students incorrectly wrote the equation "6S = P", which is a reversal of the correct equation, "S = 6P". This is a literal translation rather than a true representation of the given situation.

In a related study, Soloway, Lochhead & Clement (1982) studied the same type of problems in relation to computer programs and algebraic expressions. They found that college students were more likely to be able to express a problem (with variables) as a computer program than as an algebraic expression. They also found that students were more likely to be able to correctly "write a sentence in English" to represent a computer

program with variables than to represent a similar algebraic expression. They concluded that computer programs facilitate the understanding of variables and suggested that the use of variables in programming languages is a more "operational" process, and therefore more easily understood by students.

Some research studies have found evidence of a positive effect on students' concept and use of mathematical variables from computer programming instruction (Hart, 1982; Mayer, et al., 1986; McCoy & Burton, 1988; Oprea, 1985). Other studies have found nonsignificant results in this same area (Blume & Schoen, 1988; Salomon & Perkins, 1987).

In an interesting study, Ortiz and MacGregor (1988) studied understanding of the concept of variable among three groups: Logo programming, textbook instruction in variable, and control. They found no significant difference immediately after the treatment, but after three weeks they found that the Logo group had retained significantly more variable understanding.

Therefore, results of empirical research relating computer programming instruction and understanding of the concept of mathematical variable is unclear. While some studies have found a positive relationship, others have reported nonsignificant results. While there are no clear conclusions here, there are glimmers of possibility. Closer examination of this relationship is necessary to determine whether computer programming does or does not have the potential of improving students' understanding of variable.

The purpose of this study was to further investigate high school students' understanding of the concept of variable in a computer programming and an algebraic context.

Methods

The instrument constructed for use in this study was the ACE Translation Test, which consists of four items requiring the following translations: English expression to algebraic equation, English expression to computer program, algebraic equation to English expression, and computer program to English expression. The test contains one item for each of the four translations. No total score was generated; each item was evaluated independently. (See Appendix.)

Subjects were thirty-six high school juniors, at a large private high school. All participants had completed Algebra I and a computer literacy course which included some BASIC programming. Two classes were randomly selected and all students in those classes were given the ACE Translation Test. There were no time constraints.

Results

Results are reported as number of correct responses on each of the four items. See Table 1.

Insert Table 1 here

The data were analyzed to see if there was uniform distribution of the correct answers on the four items. Results revealed that performance on the four items was significantly different (Chi-Square (3, N = 56) = 11.57, $p < .05$). See Table 2. Thus, the four translation tasks were not equally likely to be performed correctly by the students.

Insert Table 2 here

Performance on the sum of the two computer items and on the sum of the two algebra items was significantly different, with more correct responses on the computer items (Chi-Square (1, N = 56) = 8.64, $p < .05$). See Table 3.

Insert Table 3 here

Further analysis involved comparisons of performance on the two pairs of parallel tasks. Significantly more students could correctly translate a computer program t. an English expression than could translate an algebraic equation to an English expression (Chi-Square (1, N = 32) = 8.00, $p < 0.05$). See Table 4. Although more students could translate an English expression to a computer program than to an algebraic expression, this difference was non-significant (Chi-Square (1, N = 24) = 1.50, $p > 0.05$). See Table 5.

Insert Tables 4 and 5 here

Conclusions

The results of this study indicate that high school students with experience in both algebra and computer programming could work with variables in a computer context better than in an algebraic context. When given parallel items in both a computer program and an algebraic context, their correct responses were not uniformly distributed. They were better able to translate to English the variables in the computer program than in the algebraic equation. There was no significant difference in their ability to translate an English expression to either an equation or a program.

Even though this study involved a small sample and a short test, these results suggest that there is, in fact, a relationship between computer programming and skill in using algebraic variables Participants in the study were more likely to correctly interpret a computer program than an algebraic expression when both tasks similarly used variables. The two tasks were very similar except for the context. Therefore, it seems likely that the difference is due to the context, and that students could better interpret the program because they approached it differently, i.e. in a more operational framework.

The next step, then, is to facilitate the transfer of this variable understanding ability to algebra and to other problem solving tasks. While there is no evidence of transfer of use of variables in programming to use in algebra, there is sufficient reason to further investigate this relationship and seek means to promote this transfer. Instruction must be designed to stress the similarities and to help students to bridge the gap--to make the transition from understanding variables in programming contexts to understanding them in algebraic contexts. In short, we must "teach for transfer", and a likely starting point is teaching students computer programming and then progressing to an understanding of variable in algebraic contexts.

As far as justification for including computer programming in the curriculum for all precollege students, more evidence is necessary. Further research is recommended. However, we must remember that transfer does not just "happen" (e.g. Frederikson, 1984; Hudgins, 1977). If we want students to transfer skills of understanding and using variables between computer programming and algebra, we must design instruction to highlight the similarities. We need then to study the effect of this "transfer-oriented" instruction on students' understanding of variable. As stated earlier, there do seem to be possibilities. If we are able to use computer programming to introduce and teach variables, and to facilitate transfer of this knowledge to algebra, then we may find evidence of improved general use of variables and improved problem solving skill.

REFERENCES

Blume, G. W. & Schoen, H. L. (1988). Mathematical problem-solving performance of eighth-grade programmers and nonprogrammers. Journal for Research in Mathematics Education, 19 (2), 142-156.

Clement, J. (1982). Algebra word problem solutions: Thought processes underlying a common misconception. Journal for Research in Mathematics Education, 13 (1), 16-30.

Clement, J., Lochhead, J., & Monk, G. S. (1981). Translation difficulties in learning mathematics. American Mathematical Monthly, 88, 286-290.

Clements, D. H. & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. Journal of Educational Psychology, 76 (6), 1051-1058.

Fisher, K. M. (1988). The students-and-professors problem revisited. Journal for Research in Mathematics Education, 19 (3), 260-262.

Frederiksen, N. (1984). Implications of cognitive theory for instruction in problem solving. Review of Educational Research, 54 (3), 363-407.

Hart, M. (1982). Using computers to understand mathematics. Mathematics Teaching, 52-54.

Hudgins, B. B. (1977). Learning and thinking: A primer for teachers. Itasca, IL: Erlbaum.

Johnson, D. C. & Harding, R. D. (1979). University level computing and mathematical problem-solving ability. Journal for Research in Mathematics Education, 10, 37-55.

Kantowski, M. G. (1982). Problem solving: Searching for solutions. In S. Hill (Ed.). Education in the 80's: Mathematics. Washington, D. C.: National Education Association.

Linn, M. C. (1985). The cognitive consequences of programming instruction in classrooms. Educational Researcher, 14 (5), 14-29.

Many, W. A., Lockard, J., Abrams, P. D. & Friker, W. (1988). The effect of learning to program in logo on reasoning skills of junior high school students. Journal of Educational Computing Research, 4 (2), 203-213.

Mayer, R. E. (1975). Different problem-solving competencies established with and without meaningful models. Journal of Educational Psychology, 67 (6), 725-734.

Mayer, R. E., Dyck, J. L., & Vilberg, W. (1986). Learning to program and learning to think: What's the connection? Communications of the ACM, 29 (7), 605-610.

McCoy, L. P. & Burton, J. K. (1988). The relationship of computer programming and mathematics in secondary students. Computers in the Schools, 4 (3/4), 159-166.

McCoy, L. P. & Orey, M. A. (1988). Computer programming and general problem solving by secondary students. Computers in the Schools, 4 (3/4), 151-157.

Nickerson, R. S. (1982). Computer programming as a vehicle for teaching thinking skills. Thinking: The Journal of Philosophy for Children, 4, 42-48.

Nowaczyk, R. H. (1984). The relationship of problem-solving and course performance among novice programmers. International Journal of Man-Machine Studies, 21, 149-160.

Oprea, J. M. (1985). Computer programming and mathematical thinking. Proceedings of the seventh annual meeting psychology of mathematics education--North American chapter, 212-217.

Ortiz, E. & MacGregor, S. K. (1988, April). A comparative study of a computer programming and a textbook approach in teaching the concept of variable. Paper presented at the Annual Meeting of the American Educational Research Association, New Orleans, LA.

Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. New York: Basic Books.

Pea, R. D. & Kurland, D. M. (1984). Logo programming and the development of planning skills (Tech. Rep. No. 16). New York: Bank Street College of Education, Center for Children and Technology.

Polya, G. (1957). How to solve it. Princeton: Princeton University Press.

Reed, W. M. & Palumbo, D. B. (1988). The effect of the BASIC programming language on problem-solving and computer anxiety. Computers in the Schools, 4 (3/4), 91-104.

Reed, W. M., Palumbo, D. B. & Stolar, A. L. (1988). The comparative effects of BASIC and logo instruction on problem-solving skills. Computers in the Schools, 4 (3/4), 105-118.

Salomon, G. & Perkins, D. N. (1987). Transfer of cognitive skills from programming: When and how? Journal of Educational Computing Research, 3 (2), 149-169.

Shneiderman, B. (1985). When children learn programming: Antecedents, concepts and outcomes. The Computing Teacher, 12 (5), 14-17.

Soloway, E., Lochhead, J., and Clement, J. (1982). Does computer programming enhance problem solving ability? Some positive evidence on algebra word problems. In R. J. Seidel, R. E. Anderson, and B. Hunter (Eds.) Computer literacy: Issues and directions for 1985. New York: Academic Press.

Taylor, R. P. (1980). The computer in the school: Tutor, tool, tutee. New York: Teachers College Press.

Turner, S. V. & Land, M. L. (1988). Cognitive effects of a logo-enriched mathematics program for middle school students. Journal of Educational Computing Research, 4 (4), 443-452.

Table 1.

Correct Responses on ACE Test

	NUMBER CORRECT	PERCENT CORRECT
Item 1 English/Algebra	9	25
Item 2 English/Computer	15	42
Item 3 Algebra/English	8	22
Item 4 Computer/English	24	67

N = 36

==========

Table 2.

Chi-Square Goodness-of-Fit for Uniform Distribution of Correct Responses

	Item 1	Item 2	Item 3	Item 4	Total
Observed Frequency	9	15	8	24	56

Chi-Square (3, N = 56) = 11.57, $p < .05$

Table 3.

Correct Responses on Algebra versus Computer Items

	Algebra Items	Computer Items	Total
Observed Frequency	17	39	56

Chi-Square (1, N = 56) = 8.64, p < .05

==========

Table 4.

Correct Responses on Translation TO English Expression

	From Algebra	From Computer	Total
Observed Frequency	8	24	32

Chi-Square (1, N = 32) = 8.00, p < .05

==========

Table 5.

Correct Responses on Translation FROM English Expression

	To Algebra	To Computer	Total
Observed Frequency	9	15	24

Chi-Square (1, N = 24) = 1.50, p > .05

APPENDIX

ACE (ALGEBRA, COMPUTER, ENGLISH) TRANSLATION TEST

1. Given the following statement:

   "At a recent party, for every 6 people who drank Coke, there were 11 people who drank Pepsi."

   Write an equation which represents the above statement. Use C for the number of people who drank Coke, and P for the number of people who drank Pepsi.

2. Given the following statement:

   "At a bakery, for every 4 people who ordered Chocolate Chip Cookies, there were 7 people who ordered Peanut Butter Cookies."

   Write a computer program in BASIC or Pascal which will output the number of people who ordered peanut butter cookies when supplied (via user input at the computer) with the number of people who ordered chocolate chip cookies. Use C for the number of people who ordered chocolate chip cookies, and P for the number of people who ordered peanut butter cookies.

3. Write a sentence in English that gives the same information as the following equation:

   M = 8P

   M is the number of Mercedes cars in a lot.
   P is the number of Porshe cars in a lot.

4. Program Telephones
   Input C
   T = C * 2
   Print T
   End

   For the above computer program describe in English the mathematical relationship which exists between C, the number of children, and T, the number of telephones.